НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Інститут прикладного системного аналізу

Кафедра математичних методів системного аналізу

До захисту допущено:

В.о.завідувача кафедри

________ Оксана ТИМОЩУК

«___»_____________20__ р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системний аналіз і управління»

спеціальності 124 «Системний аналіз»

на тему: «Перший прямий метод Л. С. Понтрягіна в диференціальній грі»

Виконав:
студент 4 курсу, групи КА-63
Шпильовий Ілля Вячеславович ___________

Керівник:
доцент, к.ф-м.н. Барановська Леся Валеріївна ___________

Консультант з економічний:
доцент, к.е.н. Рощина Н.В. ___________

Консультант з нормоконтролю:
доцент, к.т.н. Коваленко А. Є. ___________

Рецензент:
доцент, к.т.н. Костянтин Харченко ___________

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авторів
без відповідних посилань.
Студент _____________

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут прикладного системного аналізу

Кафедра математичних методів системного аналізу

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 124 «Системний аналіз»

Освітньо-професійна програма «Системний аналіз і управління»

ЗАТВЕРДЖУЮ

В.о.завідувача кафедри

_______ Оксана ТИМОЩУК

«___»_____________20__ р.

ЗАВДАННЯ

на дипломну роботу студенту

Шпильовому Іллі Вячеславовичу

1. Тема роботи «Перший прямий метод Л. С. Понтрягіна в диференціальній грі», керівник роботи Барановська Леся Валеріївна, кандидат фізико-математичних наук, доцент, затверджені наказом по університету від «_25_»_травня_ 20_20_ р. №_1143-с__

2. Термін подання студентом роботи 08 травня 2020 року _________

3. Вихідні дані до роботи Операційна система Windows 10/Linux OS, JDK – комплект розробника додатків на мові програмування Java, включає в себе компілятор, стандартні бібліотеки класів, приклади, документацію, виконавчу систему Java (JRE).

4. Зміст роботи

– Теоретичні поняття математичного, функціонального та теорії багатозначних відображень

– Розробка програмного продукту для дослідження диференціальної гри

– Функціонально вартісний аналіз системи

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо)

– Презентація до роботи

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Н.В., доцент		
Нормоконтроль	Коваленко А. Є., доцент		

7. Дата видачі завдання 03.04.2020 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Інструктаж з техніки безпеки	05.04.2020	Виконано
2.	Затвердження теми бакалаврської дипломної роботи (БДР). Ознайомлення зі структурою БДР згідно з Положення про випускну атестацію студентів КПІ ім. Ігоря Сікорського [Електронний ресурс] / Уклад.: В. П. Головенкін, В. Ю. Угольніков. – Київ : КПІ ім. Ігоря Сікорського, 2018. – 98 с.	05.04.2020	Виконано
3.	Ознайомлення з ДСТУ 3008-95 та стандарти Єдиної системи програмної документації (ЄСПД)	05.04.2020	Виконано

4.	Ознайомлення з державним стандартом України ДСТУ ГОСТ 7.1:2006 “Система стандартів з інформації, бібліотечної та видавничої справи. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання”	05.04.2020	Виконано
5.	Проведення дослідження за темою БДР під керівництвом керівника	12.04.2020	Виконано
6.	Завершення роботи над першим варіантом основної частини БДР	05.04.2020	Виконано
7.	Продовження роботи над експериментальною частиною БДР та програмним забезпеченням.	10.05.2020	Виконано
8.	Оформлення звіту з БДР. Складання заліку.	17.05.2020	Виконано

Студент Ілля Шпильовий

Керівник Леся Барановська

РЕФЕРАТ

Звіт до ДР: 63 с., 4 ч., 6 табл., 5 р., 3 дод., 5 джерела.

ДИФЕРЕНЦІАЛЬНА ГРА, ПЕРШИЙ ПРЯМИЙ МЕТОД Л. С. ПОНТРЯГІНА, ПРОГРАМНИЙ ПРОДУКТ.

Об'єкт дослідження – диференціальна гра переслідування-тікання двох гравців.

Мета роботи – дослідження диференціальної гри переслідування-тікання двох гравців за допомогою першого прямого методу Л.С. Понтрягіна.

Методи дослідження – моделювання гри за допомогою комп'ютерного забезпечення, а саме розрахунок часу гри за допомогою мови програмування Java.

У роботі представлені деякі математичні поняття для визначення диференціальної гри та першого прямого методу Л.С. Понтрягіна. У роботі представлений програмний продукт за допомогою якого вираховувався час модельованої гри.

ABSTRACT

Report to the GW: 63 pp., 4 hours, 6 sections, 5 years, 3 appendices, 5 sources.

DIFFERENTIAL GAME, L.S. PONTRYAGIN'S FIRST DIRECT METHOD, SOFTWARE PRODUCT.

The object of research is a differential game of chasing and running away of two players.

The purpose of the work is to study the differential game of pursuit-escape of two players using the first direct method of L.S. Pontryagin.

Research methods - computer simulation of the game, namely the calculation of game time using the Java programming language.

The paper presents some mathematical concepts for determining the differential game and the first direct method of L.S. Pontryagin. The paper presents a software product with which the time of the simulated game was calculated.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ

$\mathbb{R}^n$ – n-мірний евклідів простір.

x_i – i-та координата вектора $x, y \in \mathbb{R}^n$.

S – одинична куля.

$2^{\mathbb{R}^n}$ – множина всіх підмножин простору $\mathbb{R}^n$.

$K(\mathbb{R}^n)$ – множина всіх непорожніх компактів $\mathbb{R}^n$.

$\emptyset$ – порожня множина.

ВСТУП

Рух є невід’ємною частиною нашого буття. Протягом всієї історії людства руху приділялася увага у всіх цивілізаціях та куточках нашого світу. Люди постійно намагались пізнати та осягнути хитрощі руху та маневрів. З давніх давен люди намагалися рухатись швидше. Але не тільки швидкість руху важлива, а також і його напрямок, що суттєво ускладнює його дослідження. Однією з найважливіших частин дослідження руху є ситуація коли один об’єкт чи суб’єкт наздоганяє іншого. У світі багато прикладів подібної ситуації але з розвитком новітніх технологій постає необхідність дослідження подібних задач. У таких ситуаціях один об’єкт чи суб’єкт може мати перевагу над іншим та більше шансів досягти своєї мети. Але при цьому інший може мати перевагу у іншому. Наприклад перший учасник руху може бути досить швидким, але його ціль може мати набагато більшу маневреність. І незважаючи на те, що перший об’єкт чи суб’єкт буде дещо швидше, другий все одно зможе ухилитись. Тому перед боєм або протистояння двох або більше сил, необхідно прорахувати, як закінчиться відповідна сутичка, та виграшну стратегію із відповідними цілями. Така необхідність створила можливість для створення відповідної науки або теорії, яка називається теорією ігор. Цю теорію розвивало та продовжують підтримувати багато вчених по всьому світу, зокрема у Америці та у країнах які раніше входили до складу Радянського Союзу. При чому саму теорію ігор почали описувати ще у кінці 19 століття. А у 20 столітті теорія ігор почала сильно розвиватись завдяки холодній війні між країнами США та її союзниками проти Радянським Союзом, що дало можливості різним вченим з обох країн по різному розвивати цю теорію. Крім того теорія ігор, особливо підрозділ диференціальних ігор, дозволяє не тільки прораховувати результат бою чи протистояння між людьми чи країнами, але і враховувати різні природні явища, які важко або взагалі неможливо прорахувати в силу важких фізико-математичних викладок. Це можливість не тільки покращити економічну модель підприємства чи виробити найкращу стратегію для розвитку бізнесу, але і підвищити безпечність та зносостійкість машин та механізмів.

Метою даної роботи є дослідження диференціальних ігор переслідування-тікання задля пошуку найкращої стратегії та часу закінчення гри.

1 ДЕЯКІ МАТЕМАТИЧНІ ПОНЯТТЯ

Для визначення диференціальної гри будемо використовувати існуючі математичні поняття, які будуть визначені та описані у цьому розділі. Означення, леми та деякі доведення лем наведені та взяті з джерела [1].

1.1 Елементи скінченовимірного аналізу

Розглянемо n-мірний евклідів простір, в якому відповідно задані:

а) Скалярний добуток

$$(x,y) = \sum_{i=1}^{n} x_i y_i\,, \qquad x, y \in \mathbb{R}^n$$

б) Норма

$$\|x\| = \sqrt{(x,x)}$$

Сума векторів, як і множення вектора на число беруться стандартні, тобто покоординатно. Якщо

$$x \in \mathbb{R}^n$$

то через x_i позначається i-та координата вектора x.

Відповідно відстань визначається

$$\rho(x,y) = \|x - y\|$$

Назвемо множину

$$O_{x_0}^{\varepsilon} = \{x \in \mathbb{R}^n \mid \rho(x, x_0) \le \varepsilon\}$$

ε-окіл точки x_0 , або просто околом точки x_0 .

Введемо поняття операцій між множинами та вектором чи числом. Нехай

$$Y \subset \mathbb{R}^n, x \in \mathbb{R}^n, \alpha \in \mathbb{R}^1$$

$$x + Y = \{x \mid x + y, y \in Y\}$$

$$\alpha Y = \{x \mid \alpha y, y \in Y\}$$

Введемо таке позначення

$$S = \{x \in \mathbb{R}^n \mid \|x\| \leq 1\}$$

Це буде одинична(із радіусом 1) куля із центром у

$$\vec{0} \in \mathbb{R}^n$$

Нехай

$$X \subset \mathbb{R}^n , x \in X$$

тоді:

а) x – внутрішня точка множини X якщо

$$\exists \varepsilon \in (0 , +\infty) : \ O_x^{\varepsilon} \subset X$$

Тобто існує окіл, що повністю лежить в множині X. Відповідно множина всіх внутрішніх точок

$$int\ X = \{x \in X \mid \exists \varepsilon \in (0 , +\infty) : \ O_x^{\varepsilon} \subset X\}$$

б) x – гранична точка множини X якщо

$$\forall \varepsilon \in (0 , +\infty) : \ O_x^{\varepsilon} \cap X - \text{нескінчена}$$

Тобто будь-який окіл точки має нескінченно точок із X. Відповідно множина всіх граничних точок

$$\dot{X} = \{x \in X \mid \forall \varepsilon \in (0 , +\infty) : \ O_x^{\varepsilon} \cap X - \text{нескінчена}\}$$

в) X – відкрита якщо

$$int\ X \subset X$$

Тобто має у собі всі свої внутрішні точки.

г) X – замкнена якщо

$$\dot{X} \subset X$$

Тобто має у собі всі свої граничні точки.

д) $\overline{X}$ – замикання множини X якщо

$$\overline{X} = \dot{X} \cup X$$

Введемо операції над множинами $\mathbb{R}^n$. Нехай

$$X, Y \subset \mathbb{R}^n$$

тоді:

а) доповнення

$$X \setminus Y = \{x \mid x \in X, y \notin Y\}$$

б) об'єднання

$$X \cup Y = \{x \mid (x \in X) \vee (y \in Y)\}$$

в) перетин

$$X \cap Y = \{x \mid (x \in X) \wedge (y \in Y)\}$$

г) сума

$$X + Y = \{z \mid z = x + y, x \in X, y \in Y\}$$

д) геометрична різниця (різниця Мінковського)

$$X \dot{-} Y = \{x \mid x + Y \subset X\} = \bigcap_{y \in Y} (X - y)$$

Множина X є обмеженою якщо

$$\exists r \in (0\,, +\infty) : X \subset rS$$

Множина X є компакт якщо одночасно і замкнена і обмежена.

Нехай X – компакт. Позначимо

$$X_1 = \{x \in X \mid \forall y \in X : \ x_1 \leq y_1\}$$

$$X_2 = \{x \in X_1 \mid \forall y \in X_1 : \ x_2 \leq y_2\}$$

$$\dots$$

$$X_i = \{x \in X_{i-1} \mid \forall y \in X_{i-1} : \ x_i \leq y_i\}$$

$$\dots$$

$$X_n = \{x \in X_{n-1} \mid \forall y \in X_{n-1} : \ x_n \leq y_n\}$$

тобто X_1 – множина з найменшою 1-ю координатою серед елементів X. Для X_2 – відповідно найменша 2-га координата серед елементів X_1 і так до X_n. Очевидно, що X_n складається з одного елемента. Позначимо його через x^*. Цей елемент називають лексикографічним мінімумом множини X і позначають

$$x^* = lexmin\, X$$

Якщо множина може бути отримана зліченою сукупністю операцій додавання, доповнення, перетину, будемо називати її борелівською. Більш ширший клас множин утворюють вимірні(по Лебегу) множини, яке наведене у джерелі[2].

Розглянемо функції у просторі $\mathbb{R}^n$. Нехай

$$f: \mathbb{R}^n \to \mathbb{R}^1$$

Функція f напівнеперервна знизу(зверху) в точці x_0 якщо

$$\forall \varepsilon \in (0\,,+\infty)\ \exists \delta \in (0\,,+\infty):\ \|x - x_0\| \leq \delta\ \Rightarrow f(x) > f(x_0) - \varepsilon$$

$$(f(x) < f(x_0) + \varepsilon)$$

Що еквівалентно

$$\varliminf_{x \to x_0} f(x) = f(x_0)$$

$$(\varlimsup_{x \to x_0} f(x) = f(x_0))$$

Якщо функція напівнеперервна знизу(зверху) в кожній точці деякої множини, то вона напівнеперервна знизу(зверху) в цій множині. Якщо множина напівнеперервна знизу і зверху одночасно, то вона неперервна. Введемо ефективну множину

$$dom\, f = \{x \in \mathbb{R}^n \mid f(x) < +\infty\}$$

та множину рівня α (лебегівська множина)

$$\{x \in \mathbb{R}^n \mid f(x) < \alpha\}$$

Лема 1. Нехай X, Y, Z – компакти, функція

$$f: X \to Y$$

вимірна(борелівська), а

$$g: Y \to Z$$

борелівська(напівнеперервна зверху). Тоді функція

$$h: X \to Z, h(x) = g(f(x))$$

вимірна(борелівська). [1]

1.2 Багатозначні відображення

Розглянемо деяке відображення $F(x)$. Якщо

$$F: \mathbb{R}^n \to 2^{\mathbb{R}^n}$$

то назвемо його багатозначним відображенням. Воно ставить у відповідність вектору із $\mathbb{R}^n$ деяку множину із $\mathbb{R}^n$. На множині $K(\mathbb{R}^n)$ введемо відстань

$$dist(X,Y) = \inf\{\lambda \in (0\,,+\infty) \mid X \subset Y + \lambda S, Y \subset X + \lambda S\} =$$

$$= \max\left\{\max_{x\in X}\min_{y\in Y}\|x-y\|\,, \max_{y\in Y}\min_{x\in X}\|x-y\|\right\}$$

тоді вона стає метричним простором. Ця відстань називається хаусдорфовою метрикою. Нехай

$$F\colon \mathbb{R}^n \to 2^{\mathbb{R}^n}$$

Введемо поняття ефективної множини

$$dom\, F = \{x \in \mathbb{R}^n \mid F(x) \neq \emptyset\}$$

Багатозначне відображення

$$F\colon X \to K(\mathbb{R}^n), X \subset dom\, F$$

напівнеперервне зверху(знизу) в точці

$$x_0 \in X$$

якщо

$$\forall \varepsilon \in (0\,,+\infty)\, \exists \delta \in (0\,,+\infty) :\ \|x - x_0\| \le \delta \implies F(x) \subset F(x_0) + \varepsilon S$$

$$(F(x_0) \subset F(x) + \varepsilon S)$$

Якщо багатозначне відображення напівнеперервне зверху(знизу) в кожній точці множини X то воно називається напівнеперервним зверху(знизу) на множині X. Багатозначне відображення неперервне, якщо напівнеперервне зверху і знизу одночасно. Якщо

$$F\colon X \to K(\mathbb{R}^n)$$

є неперервним, то

$$\alpha F(x), \alpha \in \mathbb{R}$$

також неперервне.

Відображення

$$F\colon \mathbb{R}^n \to K(\mathbb{R}^n)$$

є обмеженим, якщо

$$\exists C \in \mathbb{R}\, \forall x \in \mathbb{R}^n :\ \sup\{\|y\| \mid y \in F(x)\} \le C(1 + \|x\|)$$

Твердження 1. Нехай

$$F: X \times Y \to K(\mathbb{R}^n)$$

$$X, Y \in K(\mathbb{R}^n)$$

де F – неперервне відображення.

Тоді відображення

$$G(x) = \bigcap_{y \in Y} F(x, y)$$

Напівнеперервне зверху на множині

$$X \cap \operatorname{dom} G$$

Функцію

$$f: \mathbb{R}^n \to \mathbb{R}^n$$

назвемо селектором багатозначного відображення F якщо

$$\forall x \in \mathbb{R}^n : f(x) \in F(x)$$

Багатозначне відображення

$$F: X \to K(\mathbb{R}^n)$$

будемо називати вимірним по Лебегу(Борелю) якщо:

а) $X -$ вимірне по Лебегу(Борелю)

б)

$$\forall Y \in K(\mathbb{R}^n) : \{x \in X \mid F(x) \subset Y\}$$

вимірне по Лебегу(Борелю).

Далі відображення вимірні по Лебегу будемо називати вимірними, а вимірними по Борелю – борелівськими.

Лема 2. Нехай

$$X \in K(\mathbb{R}^n), F: X \to K(\mathbb{R}^n)$$

вимірне(борелівське) відображення. Тоді селектор

$$f(x) = \operatorname{lexmin} F(x), x \in X$$

є вимірним(борелівським).

Лема 3. Нехай

$$X \in K(\mathbb{R}^n)$$

Багатозначні відображення

$$F: X \to K(\mathbb{R}^n), \qquad G: X \to K(\mathbb{R}^n)$$

є вимірними(борелівськими), функція

$$f(x,y) \in \mathbb{R}^n, x \in X, y \in G(x)$$

вимірна(борелівська) по X і неперервна по $G(x)$ для

$$\forall x \in X$$

Тоді багатозначне відображення

$$H(x) = \{y \in \; G(x) \mid f(x,y) \in F(x)\}$$

є вимірним(борелівським).

1.3 Лінійні керовані процеси

Розглянемо керований об'єкт, поведінка якого описується лінійним диференціальним рівнянням

$$\dot{x} = Ax + u, x \in \mathbb{R}^n, u \in U \qquad (1)$$

де x – n-мірний вектор фазового стану об'єкту, u – параметр керування,

$$U \in K(\mathbb{R}^n)$$

A – стала матриця порядку n. Якщо вибрано керування

$$u(\tau), \tau \geq 0$$

у вигляді вимірної функції часу із значеннями з U, то розв'язання системи (1) за допомогою формули Коші можна отримати у вигляді

$$x(t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)}u(\tau)d\tau\,, t \geq 0$$

де

$$x(t_0) = x_0$$

початковий стан процесу (1), e^{At} – фундаментальна матриця однорідної системи

$$\dot{x} = Ax$$

Нехай $A(n \times n), B(m \times m)$ – матриці. Введемо кронекеровий(прямий) їх добуток, як матрицю

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \dots & a_{1n}B \\ a_{21}B & a_{22}B & \dots & a_{2n}B \\ \dots & \dots & \dots & \dots \\ a_{n1}B & a_{n2}B & \dots & a_{nn}B \end{bmatrix}$$

розмірності $n \times m$.

Лема 4. Якщо

$$C = A \otimes I + I \otimes B$$

то

$$e^C = e^A \otimes e^B$$

1.4 Висновки

У розділі були наведені математичні означення, леми та твердження необхідні для описання елементів теорії ігор представлених у роботі. У розділі представленні елементи функціонального аналізу, та теорії багатозначних відображень.

2 ДИФЕРЕНЦІАЛЬНА ГРА

Грою називають ситуацію взаємодії двох, або більше учасників. Кожен із учасників переслідує свою ціль і зазвичай ці цілі різні, а часто протилежні, тобто виникає конфлікт. Наприклад коли перший учасник намагається наздогнати другого, який у свою чергу намагається втекти. Виходить, що якщо перший досягне своєї цілі, інший не зможе досягти свою і відповідно навпаки, якщо другому вдасться втекти, тобто зробити так, щоб перший не зміг, або не захотів його доганяти, то вийде, що «програв» перший гравець. Виходить, що одна сторона протистоїть іншій, тобто така гра є протистоянням двох сторін. Бувають такі ігри, коли на скільки виграє один з гравців, на стільки програють всі інші гравці. Таку гру називають грою з нульовою сумою.[5] Якщо виграє один з гравців, то всі інші обов'язково програють. Ігри коли один гравець намагається наздогнати іншого, який від нього втікає будемо називати іграми переслідування-тікання. Але звичайно тільки такими випадками не обмежується теорія ігор, хоча у цій роботі, в основному, будемо розглядати тільки їх.

Для представлення гри у вигляді математичної моделі використовують різні методи. Наприклад деякі найпростіші ігри можна представити у вигляді таблиці, або більш математичною мовою – матриці. По рядках можливі дії першого гравця, а по стовпчиках – другого. Відповідно на перетині записується виграш кожного. Очевидно, якщо гравців буде багато, то викладки будуть дуже громіздким, як і дослідження. Виникає проблема якщо різних ходів чи дій хоча б одного гравця нескінченно багато. Також ігри моделюють за допомогою математичних рівнянь. Якщо гра досить проста, то таке рівняння відповідно буде простим. Але зазвичай використовують системи рівнянь, якими описується відповідно стан системи у якій відбувається гра, а різними нерівностями відповідно обмеження гравців наприклад технічного характеру. Теорія ігор досить широко може застосовуватись у різних сферах життя людини, особливо в таких напрямках як економіка, воєнна справа чи інженерія, машинобудування. В економіці відповідно рівняння можуть описувати стан економіки чи деякої її частини. А для гравців важливий виграш, або наприклад кількість грошей отриманих після гри чи програш, відповідно втрачених. Для воєнної справи чи безпекового сектору будь-якої країни дуже важливі ігри на рух. В таких іграх рівняннями описується рух підрозділів чи окремих бойових одиниць, солдатів, а також їх взаємодія. Цікавою у цьому контексті є і гра переслідування-тікання. За її допомогою може бути описана ситуації переслідування ворога чи навпаки – втеча від нього. Особливо бої у небі чи космосі на ракетах. В інжнерії чи машинобудуванні рівняння описують фізичні процеси, а також стан механізмів. Перемогою є уникнення аварійних ситуацій, хоча в даній ситуації доцільніше вирахувати час гри. Очевидно, що звичайні ігри, особливо наведені вище, звичайними рівняннями описати або дуже важко або неможливо. Тому часто ігри описують диференціальними рівняннями. Такі ігри називають диференціальними іграми.[3]

У іграх важливо не тільки як діє гравець, але і те коли він так діє. Є ігри коли гравці грають по черзі відповідно виконують якусь дію і стан гри змінюється. Такі ігри називають дискретними. Час дискретної гри вимірюється в тактах або ходах. Час затрачений гравцем на один хід може взагалі не враховуватись. Але у більшості ігор, гравці «ходять» одночасно. Це означає що час гри є неперервним. Такі ігри називаються неперервними. Що дискретні, що неперервні ігри можуть бути представлені чи описані методами опису ігор. Тобто існують і диференціальні дискретні і диференціальні неперервні ігри. Останні ігри і будуть розглядатись у цій роботі.

Важливим аспектом гри є те, що гравці можуть змінювати наявний стан ситуації, або системи в якій аналізується гра. Цей фактор враховується при дослідженні гри, як керування кожного гравця окремо. Для нас, як для дослідників гри важливо те, за кого ми «граємо», тобто чий інтерес для нас в пріоритеті при дослідженні тієї чи іншої гри. Для визначення «перемоги» гравця за якого дослідник або дослідниця «грає», вводиться термінальна множина. Якщо стан досліджуваної системи є елементом цієї множини, то гра завершується, а переможець не отримує нічого, бо це лиш математичне моделювання, а не реальна ситуація. До речі, переможцем вважають гравця за якого відповідно «гралася» гра, дослідником чи дослідницею. Але в загальному дослідження не зупиняється лиш на одному гравцеві.

Як зазначалося вище кожен гравець може керувати(зазвичай собою, але ситуації бувають різні), тобто впливати на стан системи. Очевидно, що цей вплив обмежений, ба більше – для різних сторін гри цей вплив по-різному обмежений. Тобто одна із сторін може сильніше впливати в деякому аспекті. Вернемось до прикладу переслідування-тікання. Наприклад переслідувач може бути швидшим за втікача, хоча втікач може бути більш маневровим. Тобто якась із сторін може мати перевагу над іншою, але тільки в деякому аспекті. Тож досліднику чи дослідниці необхідно врахувати усі переваги і слабкості всіх сторін у всіх аспектах системи.

Важливим є те, що кожна із сторін може не тільки керувати, тобто змінювати стан чи зміну системи, а змінювати це керування відповідно до стану системи під час самої гри або ситуації. Це означає, що керування залежить від часу, тобто є функцією від часу. Якщо визначити функцію керування на всьому часу гри, то таку функцію можна назвати стратегією цього гравця на дану гру. Дослідження, або «грання» гри зазвичай включає знаходження найкращих, у деякому сенсі, стратегій для одного, декількох чи усіх гравців.

У кожній грі чи ситуації, найважливішу роль може відігравати час за який одна чи декілька сторін переможуть. Буває, що стан системи та можливості гравця не дозволяють виграти, чи взагалі гарантують поразку у грі. Тоді основною задачею такого гравця і дослідників може стати відтермінування поразки. Тому зазвичай при дослідженні чи «гранні» гри враховують також і час за який гра закінчиться навіть якщо гравець за якого «грають» дослідники гарантовано отримає поразку. Хоча у багатьох іграх час завершення не важливий і відповідно не досліджується.

Інформація викладена нижче взята переважно з джерела [1].

2.1 Задання диференціальної гри

Розглянемо задачу переслідування-тікання більш детально. Для спрощення задачі будемо розглядати гру двох гравців. Один переслідує, інший – тікає. Будемо моделювати у просторі $\mathbb{R}^n$, тому рух двох об'єктів можемо змоделювати, як рух одного багатовимірного, а гравці лиш керують цим об'єктом. Для моделювання ситуації використовується деякий керований процес. Нехай деякий рух описується рівнянням такого керованого процесу

$$\dot{z} = Az + \varphi(u, v) \tag{2}$$

де z – фазовий вектор, який відповідно описує стан системи; A – квадратна матриця порядку n, задає власне однорідне диференціальне рівняння, означає процеси які відбуваються у досліджуваній системі та з досліджуваним багатовимірним об'єктом; $\varphi: U \times V \to \mathbb{R}^n$ – неперервна по сукупності змінних функція, задає керування об'єкта усіма гравцями(в нашому випадку їх двоє, тому функція залежить від двох змінних); $U, V \in K(\mathbb{R}^n)$ – множини, які містять усі можливі стратегії керування відповідно U – для першого, тобто переслідувача, V – для другого, тобто тікача.

Під час керування кожен з гравців переслідує власні цілі. Але для закінчення гри вводиться термінальна множина. Позначимо її літерою M^*. Гра закінчується тоді, коли стан системи, в нашому випадку z, належить термінальній множині, тобто M^*. Математично можна записати умову завершення гри

$$z \in M^*$$

Ми розглядаємо гру переслідування-тікання, а саме перший гравець намагається наздогнати другого. Тому у даному типі задач можна підібрати таку термінальну множину, щоб завданням першого гравця було виконати умову завершення гри, а завданням його опонента було не дозволити цьому відбутися, або зробити так, щоб гра відбувалась нескінченно по часу. В залежності від диференціального рівняння, що описує ситуацію, можлива ситуація, що перший гравець гарантовано наздожене супротивника, або конкурента якщо перенести задачу в економічний простір. Математичною мовою це означає, що час, через який після початку завершиться гра, буде обмежений і знизу і зверху. Від матриці A та функції керування $\varphi(u, v)$, в класі задач що описується рівнянням (2), залежить чи зможе переслідувач за скінчений час завершити гру, чи тікач втече, або час завершення гри буде дорівнювати нескінченності.

Зазвичай термінальна множина M^* у задачах переслідування-тікання є циліндром у просторі $\mathbb{R}^n$ і тобто

$$M^* = M^0 + \varepsilon S \tag{3}$$

де M^0 – лінійний підпростір простора $\mathbb{R}^n$; S – шар з ортогонального доповнення до M^0 в просторі $\mathbb{R}^n$; ε – просто деяке число, яке більше за 0.

Конфліктно-керований процес визначеним рівнянням (2) називатимемо квазілінійним. Його особливість в тому, що функція керування, в загальному випадку, нелінійно залежить від параметрів керування гравців, які у свою чергу можуть бути поєднані та впливати друг на друга. Якщо $\varphi(u, v) = u - v$, то процес

$$\dot{z} = Az + u - v \tag{4}$$

є лінійним. Не зважаючи на те, що параметри керування розділені, в загальному випадку процес (4) не може бути записаний у вигляді двох незалежних лінійних процесів, одним з яких керує переслідувач, а іншим – тікач. Окремим видом процесу, що описується рівнянням (2) є лінійний процес з розподіленим рухом

$$\dot{x} = Bx + u, x \in \mathbb{R}^n$$

$$\dot{y} = Cy + v, y \in \mathbb{R}^n \tag{5}$$

Де матриці B, C – квадратні матриці порядку l, m відповідно, причому $l + m = n$.

Для перевірки нових методів дослідження ігор, використовуються декілька контрольних прикладів. Вони є класичними, укожному розглядається рух двох учасників, тобто один переслідувач та один тікач. Кожен приклад описується рівняннями по типу (5). Наведемо деякі з них.

Узагальнений контрольний приклад Л.С. Понтрягіна. Розглядається рух двох фізичних тіл в загальному випадку. Рівняння мають вигляд

$$x^{(p)} + a_1 x^{(p-1)} + \cdots + a_{p-1} x^{(1)} + a_p = u, x \in \mathbb{R}^s, u \in U$$

$$y^{(q)} + b_1 y^{(q-1)} + \cdots + b_{q-1} y^{(1)} + b_q = v, y \in \mathbb{R}^s, v \in V$$

де $x^{(i)}, y^{(j)}$ – похідні по часу порядків відповідно $i, j;\ l = p \cdot s, m = q \cdot s$. Оскільки у прикладі розглядається гра переслідування-тікання, то термінальна множина, яка у такому контексті означає, що переслідувач піймав тікача, у прикладі задається співвідношенням

$$\|x - y\| < \varepsilon$$

де ε – просто деяке число, яке більше за 0. Якщо термінальна множина задається таким чином, то називатимемо це ε-захватом. Якщо термінальна множина задається рівнянням

$$x = y$$

то називають це точним збігом геометричних координат. Якщо термінальна множина задається рівняннями

$$x = y$$

$$\dot{x} = \dot{y}$$

то називають це «м’якою посадкою», або «причалюванням». Якщо

$$p > q$$

тоді говорять, що переслідувач має більшу інерційність, але меншу маневровість. Якщо

$$p = q$$

то об’єкти мають однакову маневровість та інерційність.

Звичайний контрольний приклад Л.С. Понтрягіна. Це спрощений відносно узагальненого приклад. Моделюється другий закон Ньютона руху тіла одиничної маси під дією керуючої сили з врахуванням сили тертя яка лінійно залежить від швидкості. Описується рівняннями

$$\ddot{x} + \alpha\dot{x} = \rho u, x \in \mathbb{R}^s, \|u\| \leq 1, \alpha, \rho > 0$$

$$\ddot{y} + \beta\dot{y} = \sigma v, y \in \mathbb{R}^s, \|v\| \leq 1, \beta, \sigma > 0$$

Множини U, V – є одиничними шарами. Термінальна множина співпадає з узагальненим прикладом, але можна брати і інші термінальні множини такі як точний збіг координат.

Приклад коливальний процес. Моделювання коливання математичного маятника. Описується рівняннями

$$\ddot{x} + \alpha^2 x = u, x \in \mathbb{R}^s, \|u\| \leq \rho$$

$$\ddot{y} + \beta^2 y = v, y \in \mathbb{R}^s, \|v\| \leq \sigma$$

де числа α, β – задають амплітуди коливань математичних маятників, а кожна із однорідних систем задає періодичний рух кожного із маятників. Термінальна множина задається рівнянням

$$x = y$$

або вимагає точний збіг координат.

Якщо процес описується диференціальними рівняннями з другою похідною, то називається він процесом другого порядку. Аналогічно з диференціальними рівняннями вищих порядків.

Приклад «хлопчик і крокодил». Задача моделює ситуацію коли перший об'єкт, який переслідує, має більшу швидкість, але при цьому меншу маневровість відносно другого, який тікає. Першого об'єкта умовно називають «крокодилом», а другого – «хлопчиком». Процес описується рівняннями

$$\ddot{x} = u, x \in \mathbb{R}^s, \|u\| \leq 1$$

$$\dot{y} = v, y \in \mathbb{R}^s, \|v\| \leq 1$$

Обмежений радіус першого гравця вказує на низьку маневровість «крокодила». «Хлопчик» хоч і повільніший, зате значно маневровіший. Термінальна множина визначається нерівністю

$$\|x - y\| \leq \varepsilon$$

Тобто достатній ε-захват «хлопчика», щоб «крокодил» переміг.

Приклад постійний рух. Процес задається простим рухом, якщо описується відповідними рівняннями

$$\dot{x} = u, x \in \mathbb{R}^s, \|u\| \le a$$

$$\dot{y} = v, y \in \mathbb{R}^s, \|v\| \le 1$$

Процес є першого порядку. Кожен із об'єктів однаково неінерційний та має необмежену маневровість. Термінальна множина вимагає лиш ε-захват, тобто визначається нерівністю

$$\|x - y\| \le \varepsilon$$

Кожен із процесів наведених у прикладах вище, може бути через відповідну заміну змінних записаний рівняннями типу(4), (5).

2.2 Перший прямий метод Л.С. Понтрягіна

Усіх використані нижче леми, теореми, зауваження, твердження наведені у джерелі (1). Розглянемо квазілінійний конфліктно-керований процес

$$\dot{z} = Az + \varphi(u, v), z \in \mathbb{R}^n, u \in U, v \in V$$

$$U, V \in K(\mathbb{R}^n)$$

$$\varphi: U \times V \to \mathbb{R}^n \tag{6}$$

де A – стала квадратна матриця порядку n, яка визначає некеровану частину процесу; φ – функція, неперервна по сукупності параметрів керування, кожного з гравців, розглядаємо гру з двома.

Термінальна множина є циліндричним та задається рівнянням

$$M^* = M^0 + M$$

де M^0 – лінійний підпростір $\mathbb{R}^n$, M – непорожній компакт із ортогонального доповнення до M^0 у просторі $\mathbb{R}^n$, яке позначимо як L.

Позначимо оператор ортогонального проектування з $\mathbb{R}^n$ у L, як π, а фундаментальну матрицю однорідного рівняння

$$\dot{z} = Az$$

як e^{At}. Початковий стан позначимо z^0. Введемо позначення

$$\varphi(v) = \{\varphi(u, v) \mid u \in U\}, v \in V$$

$$W(t, v) = \pi e^{At} \varphi(v)$$

$$W(t) = \bigcap_{v \in V} W(t, v), t \geq 0$$

Умова Понтрягіна.

$$\forall t \geq 0 : W(t) \neq \emptyset.$$

Зауваження 1.

$$\varphi(u, v) = u - v \Leftrightarrow W(t) = \pi e^{At} U \dot{-} \pi e^{At} V$$

З умови Понтрягіна маємо

$$\operatorname{dom} W(t) = [0; +\infty)$$

Тоді оскільки $W(t, v)$ неперервно на множині $[0; +\infty) \times V$, то згідно Твердження 1, $W(t)$ – напівнеперервне зверху, звідки втікає, що і борелівське. Тому згідно Леми 2 існує хоча б один борелівський селектор

$$\gamma(\tau) \colon \mathbb{R}^n \to \mathbb{R}^1$$

$$\gamma(\tau) \in W(t), t \geq 0$$

Введемо позначення

$$\Gamma = \{\gamma(t) \mid \gamma(t) \in W(t), t \geq 0\}$$

Введемо функцію Л. С. Понтрягіна

$$P \colon \mathbb{R}^n \to \mathbb{R}^1$$

$$P(z) = \inf \left\{ t \geq 0 \mid \pi e^{At} z \in M - \int_0^t W(x) dx \right\}$$

якщо множина, що визначена у фігурних дужках порожня, тобто

$$\forall t \geq 0 : \pi e^{At} z \notin M - \int_0^t W(x) dx$$

то покладемо

$$P(z) = +\infty$$

Лема 4. Нехай для керованого процесу(6), виконується умова Понтрягіна. Якщо

$$\exists z : P(z) < +\infty$$

Тоді

$$\exists t_0 \in \left\{ t \geq 0 \mid \pi e^{At} z \in M - \int_0^t W(x)dx \right\} : P(z) = t_0$$

або існує нижня грань у функції Понтрягіна.

Теорема Понтрягіна. Нехай для керованого процесу виконується умова Понтрягіна. Якщо

$$\exists z^0 : P(z^0) < +\infty$$

Тоді стан системи може бути переведений з z^0 у термінальну множину в момент $P(z^0)$.

Доведення. Через лему (4), маємо

$$\pi e^{AP(z^0)} z^0 \in M - \int_0^{P(z^0)} W(P(z^0) - t)dt$$

Тоді

$$\exists m \in M$$

$$\exists \gamma(t) \in \Upsilon$$

$$\pi e^{AP(z^0)} z^0 = m + \int_0^{P(z^0)} \pi e^{A(P(z^0)-t)} \gamma(P(z^0) - t)dt$$

Розглянемо багатовимірне відображення

$$U(t, v) = \left\{ u \in U \mid \pi e^{A(P(z^0)-t)} \varphi(u, v) - \gamma(P(z^0) - t) \right\}, t \in [0, P(z^0)], v \in V$$

Згідно леми 3 воно борелівське по (t, v). Селектор

$$u(t, v) = \operatorname{lexmin} u(t, v(t)), t \in [0, P(z^0)]$$

є борелівським згідно леми 2. Керування переслідувача має бути

$$u^*(t) = u(t, v(t))$$

де $v(t) \in V, t \in [0, P(z^0)]$ – деяке довільне керування тікача. Згідно леми 1, воно є вимірною функцією від часу. Підставивши корінь диференціального рівняння Коші у рівняння викладені вище маємо

$$\pi z\big(P(z^0)\big) = \pi e^{AP(z^0)} z^0 + \int_0^{P(z^0)} \pi e^{A(P(z^0)-t)} \varphi(u(t), v(t)) dt$$

Доведено.

Цю теорему та метод побудови керування, що наведений у доведенні називають першим прямим методом Л.С. Понтрягіна.

Для представлення першого прямого методу Л.С. Понтрягіна, дослідимо один з контрольних прикладів цим методом. Досліджувати будемо контрольний приклад Л.С. Понтрягіна.

2.3 Контрольний приклад Л.С. Понтрягіна

Розглянемо гру переслідування-тікання. Необхідно знайти час закінчення гри. Будемо досліджувати із сторони переслідувача, тому необхідно визначити його стратегію. Маємо конфліктно-керований процес

$$\ddot{x} + a\dot{x} = ru, x \in \mathbb{R}^s, \|u\| \le 1, a, r > 0$$

$$\ddot{y} + b\dot{y} = cv, y \in \mathbb{R}^s, \|v\| \le 1, b, c > 0$$

$$u \in U, v \in V$$

Введемо заміну змінних

$$z = column(z_1, z_2, z_3)$$

$$z_1, z_2, z_3 \in \mathbb{R}^n$$

$$z_1 = x - y$$

$$z_2 = \dot{x}$$

$$z_3 = \dot{y}$$

Термінальна множина задаємо, як точний збіг координат. Тобто

$$M^* = \{z \mid z_1 = 0\}$$

$$M^0 = \{z \mid z_1 = 0\}$$

$$L = \{z \mid z_2 = 0, z_3 = 0\}$$

$$M = \{z \mid z_2 = 0, z_3 = 0\}$$

Маємо диференціальне рівняння

$$\dot{z} = Az + u - v$$

$$\dot{z}_1 = z_2 - z_3$$

$$\dot{z}_2 = -az_2 + ru$$

$$\dot{z}_3 = -bz_3 + cv$$

Визначаємо множини керування

$$U = \left\{ \begin{pmatrix} 0 \\ ru \\ 0 \end{pmatrix} \mid \|u\| \le 1 \right\}$$

$$V = \left\{ \begin{pmatrix} 0 \\ 0 \\ -cv \end{pmatrix} \mid \|v\| \le 1 \right\}$$

Отримуємо матрицю

$$A = \begin{pmatrix} 0 & 1 & 1 \\ 0 & -a & 0 \\ 0 & 0 & -b \end{pmatrix} \otimes \mathbb{I}$$

$$\mathbb{I} \subset \mathbb{R}^s$$

Отримуємо ортогональний проектор та фундаментальну матрицю однорідного диференціального рівняння

$$\pi = \begin{pmatrix} 0 & 1 & 1 \\ 0 & -a & 0 \\ 0 & 0 & -b \end{pmatrix} \otimes \mathbb{I}$$

$$e^{At} = \begin{pmatrix} 1 & \dfrac{1 - e^{-at}}{a} & \dfrac{e^{-bt} - 1}{b} \\ 0 & e^{-at} & 0 \\ 0 & 0 & e^{-bt} \end{pmatrix} \otimes \mathbb{I}$$

Знайдемо багатозначне відображення з умови Понтрягіна

$$W(t) = \pi e^{At} U \dot{-} \pi e^{At} V$$

$$\pi e^{At} U = \frac{1 - e^{-at}}{a} rS$$

$$\pi e^{At}V = \frac{1-e^{-bt}}{b}cS$$

$$W(t) = \frac{1-e^{-at}}{a}rS \dot{-} \frac{1-e^{-bt}}{b}cS = \left(\frac{1-e^{-at}}{a}r - \frac{1-e^{-bt}}{b}c\right)S = w(t)S$$

Це відображення є одиничною кулею, радіус якої визначається $w(t)$. Оскільки

$$\forall t \geq 0 : w(t) \in \mathbb{R}^1$$

то

$$\forall t \geq 0 : W(t) \neq \emptyset$$

тобто умова Понтрягіна виконується завжди. Можемо скористатись теоремою Понтрягіна. Візьмемо селектор тотожний нулю, тоді

$$U(t,v) = \left\{u \in U \mid \pi e^{A(P(z^0)-t)}(u-v) = 0\right\}$$

Отримаємо

$$u^*(t) = v(t)$$

Шукаємо час закінчення гри

$$P(z^0) = \min\left\{t \geq 0 \mid \pi e^{At}z^0 \in M - \int_0^t W(x)dx\right\} =$$

$$= \min\left\{t \geq 0 \mid \pi e^{At}z^0 \in -\int_0^t w(x)Sdx\right\} =$$

$$= \min\left\{t \geq 0 \mid \pi e^{At}z^0 \in \int_0^t w(x)Sdx\right\} =$$

$$= \min\left\{t \geq 0 \mid \pi e^{At}z^0 \in \int_0^t w(x)dx\, S\right\} =$$

$$= \min\left\{t \geq 0 \mid (z_1^0 + \frac{1 - e^{-at}}{a} z_2^0 - \frac{1 - e^{-bt}}{b} z_3^0) \in \right.$$

$$\left. \in \int_0^t \left(\frac{1 - e^{-ax}}{a} r - \frac{1 - e^{-bx}}{b} c\right) dx\, S\right\} =$$

$$= \min\left\{t \geq 0 \mid (z_1^0 + \frac{1 - e^{-at}}{a} z_2^0 - \frac{1 - e^{-bt}}{b} z_3^0) \in \right.$$

$$\left. \in \int_0^t \left(\frac{1 - e^{-ax}}{a} r - \frac{1 - e^{-bx}}{b} c\right) dx\, S\right\} =$$

$$= \min\left\{t \geq 0 \mid (z_1^0 + \frac{1 - e^{-at}}{a} z_2^0 - \frac{1 - e^{-bt}}{b} z_3^0) \in \right.$$

$$\left. \in \left(\left(\frac{r}{a} - \frac{c}{b}\right) t + (1 - e^{-at}) - (1 - e^{-bt})\right) S\right\} =$$

$$= \min\left\{t \geq 0 \mid \left\| z_1^0 + \frac{1 - e^{-at}}{a} z_2^0 - \frac{1 - e^{-bt}}{b} z_3^0 \right\| \leq \right.$$

$$\left. \leq \left|\left(\frac{r}{a} - \frac{c}{b}\right) t + (1 - e^{-at}) - (1 - e^{-bt})\right|\right\}$$

Наступним кроком дослідження є комп’ютерне моделювання. Воно відбувалося за допомогою програмного продукту описаного нижче.

2.4 Висновки

У розділі представлені елементи теорії ігор необхідні для формулювання першого прямого методу Л.С. Понтрягіна. Також наведений приклад дослідження диференціальної гри. Також наведені деякі базові елементи теорії ігор для більшого розуміння викладок у розділі.

3 ПРОГРАМНИЙ ПРОДУКТ

Метою моделювання був розрахунок часу закінчення гри із визначених параметрів та початкового положення для контрольного прикладу Понтрягіна. Програмний продукт є консольною програмою, яка обробляє вміст вхідного файлу, а результат записує у вихідний файл. Програма виконана з використанням мови програмування Java.

Ця мова програмування має багато переваг для вибрання її. Один з них – це кроссплатформеність. Персональні комп'ютери, смартфони та термінали, які підтримують запуск консольних додатків(а таких більшість) та JRE. Зможуть скористатись програмним продуктом.

Суттєвим недоліком є відсутність графічного інтерфейсу. Але цей недолік легко усунути завдяки гнучкості та поширення Java. Також важливим недоліком є те, що програмний продукт проводить моделювання для занадто обмеженого класу задач. Цей недолік теж можна компенсувати гнучкістю Java але мова програмування Java поступається іншим мовам у вирахуванні складних математичних викладок, які обов'язково доведеться застосовувати при розширені функціоналу програми.

3.1 Мова програмування Java

Мова програмування Java є компільованою мовою програмування. Написаний вихідний код компілюється у виконуваний файл. Але цей файл виконує на пряму не операційна система комп'ютера, а спеціальне програмне забезпечення, яке називається середовищем виконання Джава, скорочено JRE. Якщо об'єднати JRE з засобами компіляції коду вийде JDK. Далі називатимемо її – джава машина, а виконувані файли джава – джава-файли.

Для виконання джава-файлів необхідно на комп'ютер чи термінал спочатку встановити цю віртуальну машину.

Оскільки для багатьох операційних систем та терміналів існують джава машини, досягається кросплатформеність. Це дозволяє використовувати програму на багатьох приладах.

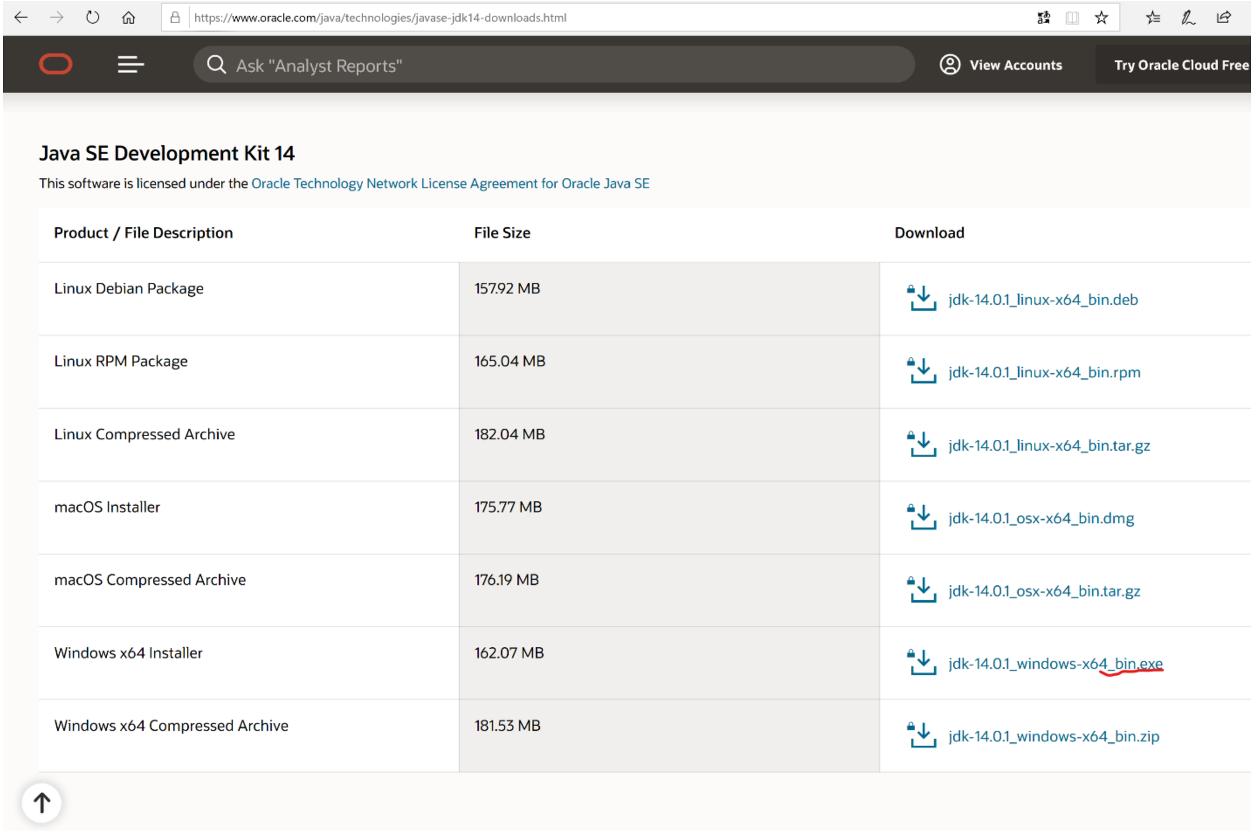
Суттєвими недоліками є те, що крім власне програми комп’ютеру доводиться ще й виконувати різні інструкції джава машини, що знижує швидкість виконання алгоритму розрахунку. Особливо це видно при важких та великих обсягах математичних розрахунків.

Важливо, що розробкою мови Java на сьогодні займається компанія Oracle. Тому бажано скачувати JRE з офіційного сайту Oracle.

3.2 Встановлення JRE на Windows 10

Завантаження інсталятора для JRE на Windows дещо важче ніж для JDK, тому інсталюємо його, а функціонал тільки збільшиться. Для встановлення JDK на Windows необхідно спочатку завантажити інсталятор Windows. Спершу перейдемо на офіційний сайт Oracle у місце завантаження JDK.

На сайті є інсталятори та архіви для різних систем та платформ. Важливо скачати інсталятор точно для необхідної платформи. На рисунку 3.1 червоним позначено розширення, яке використовується для інсталяторів.



Java SE Development Kit 14

This software is licensed under the Oracle Technology Network License Agreement for Oracle Java SE

Product / File Description	File Size	Download
Linux Debian Package	157.92 MB	jdk-14.0.1_linux-x64_bin.deb
Linux RPM Package	165.04 MB	jdk-14.0.1_linux-x64_bin.rpm
Linux Compressed Archive	182.04 MB	jdk-14.0.1_linux-x64_bin.tar.gz
macOS Installer	175.77 MB	jdk-14.0.1_osx-x64_bin.dmg
macOS Compressed Archive	176.19 MB	jdk-14.0.1_osx-x64_bin.tar.gz
Windows x64 Installer	162.07 MB	jdk-14.0.1_windows-x64_bin.exe
Windows x64 Compressed Archive	181.53 MB	jdk-14.0.1_windows-x64_bin.zip

Рисунок 3.1 – сайт завантаження

Якщо скачати архів замість інсталятора, то встановлення буде набагато важче, хоча в деяких випадках деяким професіоналам краще архів.

Після натискання на посилання для закачування доведеться погодитись з ліцензійною згодою. JDK поставляється безкоштовно, що ще більше розширює область застосування програмного забезпечення.

Для погодження з угодою необхідно поставити галочку в квадратик як зображено на рисунку 3.2. Потім достатньо натиснути на велику зелену кнопку під квадратиком.

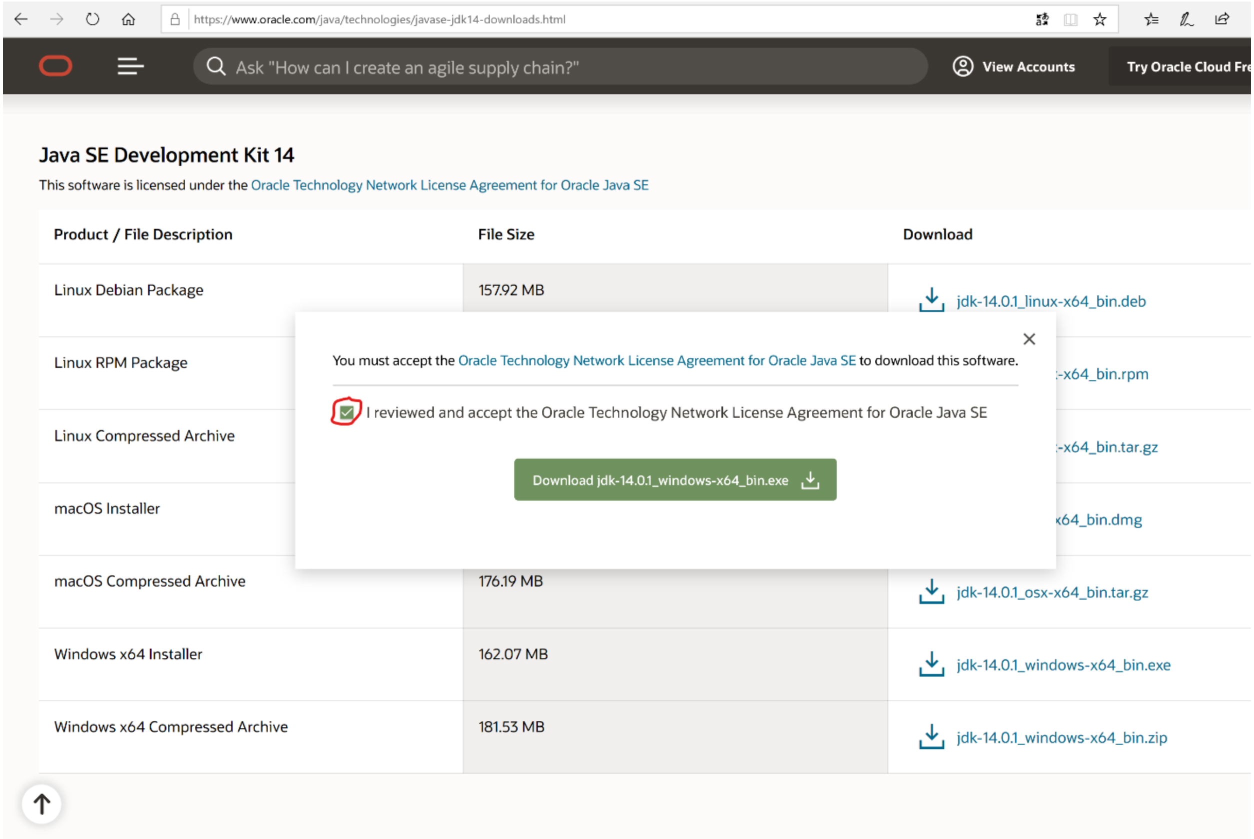


Рисунок 3.2 – ліцензійна угода

Почнеться завантаження інсталятора. Після закінчення достатньо його відкрити і слідувати інструкціям під час інсталяції. Важливо під час інсталяції запам’ятати шлях встановлення JDK. Для цього необхідно під час інсталяції скористатись опцією зміни місця призначення. Запам’ятати шлях. І нічого не змінюючи продовжити інсталяцію.

Для роботи JDK через командний рядок у Windows 10 необхідно прописати шлях до папки ./bin, яка знаходиться у місці інсталяції JDK. Зазвичай достатньо перейти у цю папку використовуючи Провідник, та скопіювати шлях.

Треба один раз натиснути лівою клавішею миші у місце показане на рисунку 3.3. Потім скопіювати вибране.

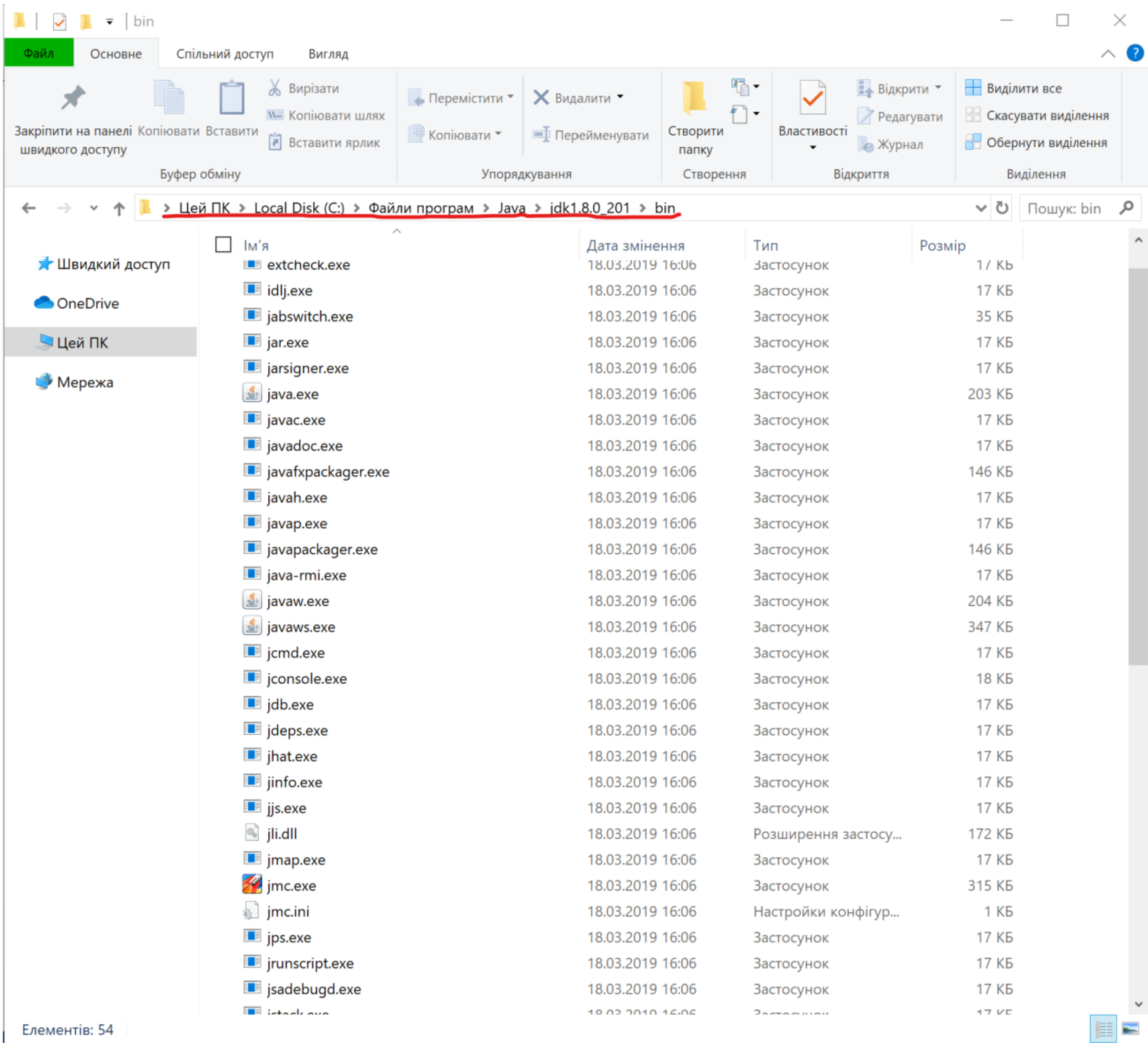


Рисунок 3.3 – шлях

Далі треба записати цей шлях у системну змінну Path.

Треба перейти до Панелі керування, зайти у розділ Система, потім у підрозділ Система, натиснути на посилання Додаткові настройки системи.

З'явиться діалогове вікно, де у вкладці Додатково натиснути кнопку Змінні оточення. У новому діалоговому вікні, у розділі Системні змінні знайти змінну Path та двічі натиснути на неї лівою клавішею миші.

У ще одному діалоговому вікні кнопку Створити і засобами операційної системи вставити шлях до папки ./bin. Зберегти усі зміни в усіх діалогових вікнах. Тепер можна запускати програму.

3.3 Використання програмного забезпечення

Програмне забезпечення поставляється у джава-архіві. Для роботи програми необхідний вхідний файл у якому записані вхідні дані, а саме параметри та початковий стан. У вхідному файлі по черзі записані числа. Вхідний файл має називатися тільки “datain.txt”, але це можна не дуже важко змінити засобами JDK. Між числами можуть бути пробіли, символи переносу рядка чи табуляції, для програми це не має особливого значення, хоча чим менше символів, тим краще. Тому можна записати числа структуровано, щоб легше сприймати, або усі числа через пробіл, якщо файл генерується іншою програмою. У самому вхідному файлі дані записуються по порядку, параметрами з розділу 2.3 показаний порядок

$$s\ r\ c\ a\ b\ z_{11}^0 \dots z_{1s}^0\ z_{21}^0 \dots\ z_{2s}^0 z_{31}^0 \dots z_{3s}^0$$

Де z_{is}^0 – s-та координата вектору z_i^0.

Таким чином записується одне моделювання прикладу. Якщо записати у вхідний файл декілька таких моделювань, то програма їх порахує.

У вихідний файл записуються результати усіх моделювань в стовпчик, тобто через символ переносу каретки. Вихідний файл називається “dataout.txt”.

Для виконання розрахунку необхідно через консоль зайти у папку де містяться файли програми та вхідний файл, та виконати команду на рисунку 3.4. Перше виконання успішне і вихідний файл створено з результатом розрахунку. А в другому випадку немає вхідного файла у папці з програмою.

Рисунок 3.4 – запуск

У додаток А наведений вихідний код кожного класу, а у додатку Б наведені приклади вхідного та вихідного робочих файлів.

3.4 Висновки

У розділі розглянуто метод запуску та використання програмного продукту. Крім цього методу програму можливо використовувати іншими методами. Зазначеним методом не обов’язково обмежуватись для використання програмного продукту.

4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

4.1 Постановка задачі техніко-економічного аналізу

У роботі застосовується метод функціонально-вартісного аналізу для проведення техніко-економічного аналізу розробки програмного продукту для моделювання диференціальної гри.

Технічні вимоги до продукту наступні:

- програмний продукт повинен функціонувати на сучасних комп’ютерах з доступом до Інтернету;

- забезпечувати високу швидкість прийняття рішень;

-забезпечувати максимальну відсутність похибок, що призводять до втрат капіталу;

-забезпечувати зручність і простоту встановлення на будь-яку апаратну систему та поєднання з іншими програмними системами;

-передбачати мінімальні витрати на впровадження програмного продукту.

4.2 Обґрунтування функцій та параметрів програмного продукту

Функція F0 — розробка програмного продукту, що приймає на вхід дані та виводить рекомендації для прийняття рішення щодо цінних паперів. Виходячи з цього, можна виділити наступні основні функції ПП:

F_1— вибір мови програмування: мова програмування C++; б) мова програмування Java;

F_2 — походження математичних бібліотек: а) існуючі; б) власна;

F_3— алгоритм роботи: а) власний алгоритм; б) існуючий.

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1). За допомогою цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій(таблиця 4.1).

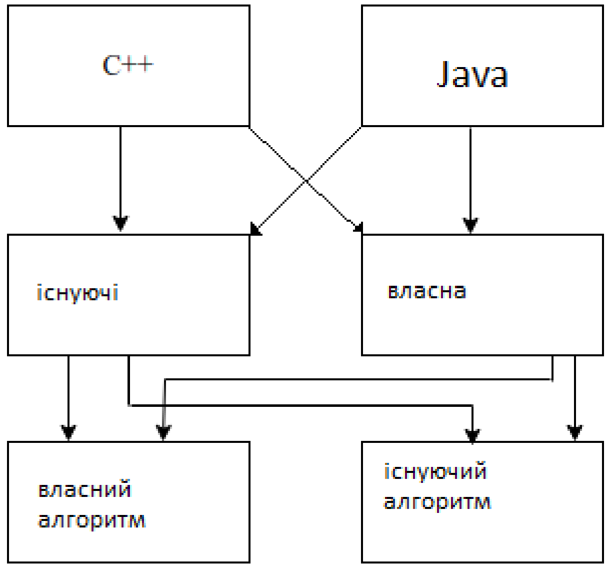


Рис 4.1 – Морфологічна карта

Таблиця 4. 1 — Позитивно негативна матриця

Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	*А*	Висока швидкодія, оптимізація під різні платформи	Висока вартість розробки, складність введення нових алгоритмів
	Б	Нижча вартість, відносна легкість проведення дослідів	Низька швидкодія
F2	*А*	Кросплатформеність, полегшений процес завантаження нових даних	Складність при тестуванні
	Б	Нижча вартість та вища швидкість розробки	Більш вузький функціонал.
	А	Велика кількість постачальників даних	Низька зручність в користуванні

F3	*Б*	Зручний інтерфейс отримання даних	Висока вартість розробки та підтримки.

За аналізом позитивно-негативної матриці можна зробити висновок, що при розробці програмного продукту деякі варіанти реалізації функцій потрібно відкинути, бо вони не відповідають поставленій перед програмним продуктом меті.

Функція *F1:*

Оскільки Java дозволяє легко опрацьовувати дані, то розглядаємо лише цей варіант.

Функція *F2:*

Вартість програмного продукту залежить від швидкості роботи, тому варіант а) не розглядається.

Функція *F3:*

Так як дані у джерелах, що розглядаються, постійно оновлюються, тому розглядаються обидві варіанти.

Таким чином, будемо розглядати такий варіант реалізації ПП:

1. F1a — F2б — F3a
2. F1a — F2б — F3б.

Функція F3 залежить від двох параметрів X3 і X4(опис даних параметрів наведено в таблиці 4.2).

Для оцінювання функцій обрана система параметрів (табл.. 4.2), де визначені гірші, середні і кращі значення параметрів обираємо на основі вимог замовника й умов експлуатації .

Таблиця 4.2. - Основні параметри програмного продукту

Назва Параметра	**Умовні позначення**	**Одиниці виміру**	**Значення параметра**		
			гірші	середні	кращі
Швидкодія мови програмування	X1	нс/Оп	350	120	100

Об’єм пам’яті для коректної роботи	X2	Кб	640	320	80
Час оволодіння теорією	X3	г	3000	1200	220
Потенційний об’єм програмного коду	X4	кількість рядків коду	360	240	120

Графічні характеристики параметрів на Рисунку 4.2

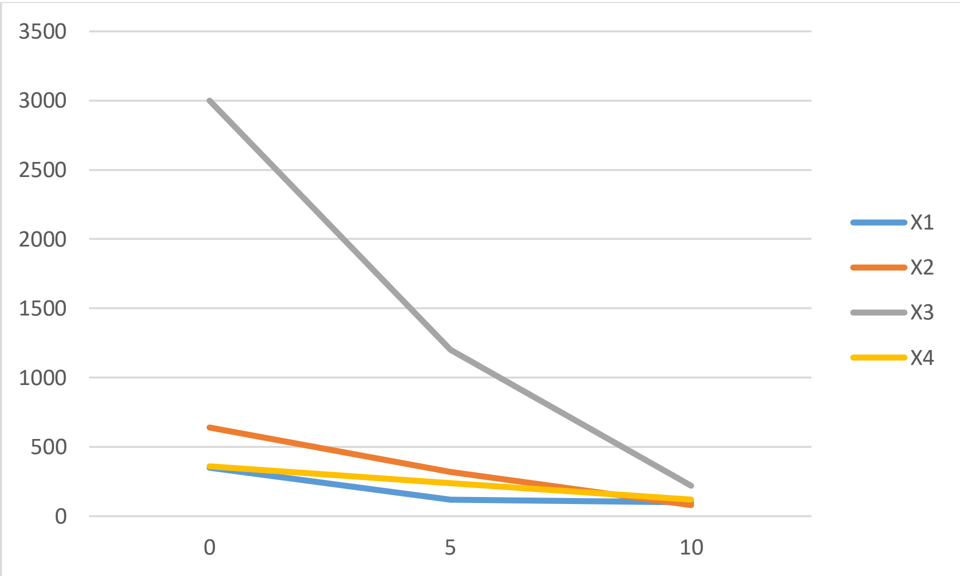


Рисунок 4.2 Графіки зміни параметрів X1, X2, X3, X4.

Таблиця 4.3 — Результати ранжування параметрів

Позначення параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів Ri	Відхилення Δi	$\Delta^2{}_i$
		1	2	3	4	5	6	7			
X1	нс/Оп	3	2	4	2	1	1	2	12	-5.5	30.25
X2	Мб	2	1	1	1	2	1	1	9	-8.5	72.25
X3	мс	3	3	4	3	4	4	3	24	6.5	42.25
X4	кількість рядків коду	4	4	3	4	3	3	4	25	7.5	56.25
Разом		10	10	10	10	10	10	10	70	0	201

⍰0

Коефіцієнт узгодженості $W = \frac{12*S}{N^2(n^3-n)} = \frac{12*201}{49*(64-4)}$ $.82 > 0.67.$ Ранжування є достовірним згідно нерівності.

Таблиця 4.4 - Попарне оцінювання експертів

Параметри	**Експерти**							**Кінцева оцінка**	**Числове значення**
	1	2	3	4	5	6	7		
X1 і X2	>	<	<	<	>	<	<	<	0.5
X1 і X3	>	>	>	>	>	>	>	>	1.5
X1 і X4	>	>	>	>	>	>	>	>	1.5
X2 і X3	>	>	>	>	>	>	>	>	1.5
X2 і X4	>	>	>	>	>	>	>	>	1.5
X3 і X4	>	>	<	>	<	<	>	<	1.5

Таблиця 4.5 - Розрахунок вагомості параметрів

Параметри x_i	**Параметри x_j**				**Перша ітерація**		**Друга ітерація**		**Третя ітерація**	
	1	2	3	4	b_i	K_b	b_i^1	K_{bi}	b_i^2	K_{bi}
X1	1	2	1	2	5	0	16	0.275	59.125	0.274
X2	1.5	1	2	2	6	0	21	0.36	77.875	0.361
X3	0.5	2	1	1	4	0	12	0.208	44.875	0.207
X4	0.5	2	1	1	3	0	9	0.157	34.125	0.158
Всього:					16	1	59	1	216	1

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	**Варіант реалізації**	**Параметри –**	**Абсолютне значення параметра**	**Бальна оцінка параметра**	**Коефіцієнт вагомості параметра**	**Коефіцієнт рівня якості**
F1	А	X1	1	10	0.274	2.74
F2	Б	X2	8	10	0.361	3.61
F3	А	X3	2000	3	0.207	0.621
		X4	1500	2	0.158	0.316
	Б	X3	220	10	0.207	2.07

		X4	900	10	0.158	1.58

$K_{K1} = 2.74 + 3.61 + 0.621 + 0.316 = 7.287$

$K_{K2} = 2.74 + 3.61 + 2.07 + 1.58 = 10$

Перший варіант виявився кращим, оскільки коефіцієнт технічного рівня має найбільше значення.

4.3. Економічний аналіз варіантів розробки програмного продукту

Усі варіанти включають в себе наступні завдання:

1. Розробка методів підготовки даних для їх використання в ПП:

Для першого варіанту реалізації використовуємо нормалізацію;

Для другого варіанту – цифрову фільтрацію

2.Розробка проекту програмного продукту;

3.Розробка програмної оболонки;

Для першого завдання((при реалізації варіанту 1А алгоритм групи складності 3 , ступінь новизни В, вид використаної інформації — НДІ) $T_P = 12$, $K_П = 1.26$, $K_{СК} =1$, $K_{СТ.М} = 1$

$T_1^1 = 12 * 1.26 = 15.12$ людино-днів

Для першого завдання (при реалізації варіанту 1Б, (алгоритм групи складності 1, ступінь новизни А, вид використаної інформації — НДІ))$T_P =27$,

$K_П =1.26$ для НДІ, $K_{СК} = 1$, $K_{СТ.М} =1.6$.

$T_1^2 = 27 * 1.26 * 1.6 = 54.432$ людино-днів

Для другого завдання (ступінь новизни А, група складності алгоритму 1) $T_р = 90$ людино-днів, $K_П = 1.6$, $K_{СК} = 1$, $K_{СТ} = 0.8$.

$$T_2 = 90*1.6*0.8 = 115.2 \text{ людино-днів.}$$

Для третього завдання(ступінь новизни Б, група складності алгоритму 3) $T_р = 28$ людино-днів, $K_П = 0.7$, $K_{СТ} = 0.8$:

$$T_3 = 28*0.7*0.8 = 15.68 \text{ людино-днів}$$

Трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми

$$T_I = (15.12 + 115.2 + 15.68) \cdot 8 = 1168 \text{ людино-годин};$$

$$T_{II} = (54.432 + 115.2 + 15.68\,) \cdot 8 = 1482.496 \text{ людино-годин};$$

Найбільш високу трудомісткість має варіант II.

В розробці бере участь один фінансовий інженер з окладом 28000 грн.

Зарплата за годину:

$$C = \frac{28000}{1*21*8} = 166.67\text{грн.}$$

Зарплата розробників

I. $C_{ЗП}$ = 166.67*1168*1.2 = 233605 грн.

II. $C_{ЗП}$ = 166.67*1482.496*1.2 = 296505 грн.

Відрахування на соціальний внесок становить 22,0%:

I . $C_{ВІД}$= 233605 * 0.22 = 51393 грн.

II. $C_{ВІД}$= 296505 * 0.22 = 65231 грн.

Обчислимо витрати на оплату однієї машино-години. Оскільки ЕОМ обслуговує один інженера з окладом 28000 грн., з коефіцієнтом зайнятості 0.2, то для однієї машини маємо:

$C_Г$ = 12*М*$К_З$ = 12 * 28000*0.2 = 67200 грн.

З урахуванням додаткової заробітньої плати:

$C_{ЗП}$ =$C_Г$*(1 + $К_З$)= 67200*(1+0.2) = 80640 грн.

Відрахування на соціальний внесок:

$C_{ВІД}$ = $C_{ЗП}$ * 0.22 = 80640 * 0.22 = 17740.8 грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 140000 грн.

$C_А$ = $К_{ТМ}$ * $К_А$*$Ц_{ПР}$ = 1.15*0.25*140000 = 40250 грн.,

Витрати на ремонт та профілактику:

$C_Р$ = $К_{ТМ}$*$Ц_{ПР}$*$К_Р$ = 1.15 * 140000*0.05 = 8050 грн.

Ефективний годинний фонд часу ПК:

$T_{ЕФ} = (Д_К — Д_В — Д_С — Д_Р) * t_З * К_В = (365 — 104 — 8 — 16)* 8 * 0.9 =$ 1706.4 годин

Витрати на оплату електроенергії (з урахуванням ПДВ):

$С_{ЕЛ} = Т_{ЕФ} * N_С*К_З*Ц_{ЕН} = 1706.4 * 0.2 * 2 * 1.46255 * 1.75 = 1746.99$ грн.,

Накладні витрати

$С_Н = Ц_{ПР} * 0.67 = 140000*0.67 = 93800$ грн.

Річні експлуатаційні втрати:

$С_{ЕКС} = 80640 + 17740.8 + 40250 + 8050 + 1746.99 + 93800 = 242227.79$ грн.

Собівартість однієї машино-години ЕОМ дорівнює:

$С_{М-Г} = С_{ЕКС} / Т_{ЕФ} = 242227.79 / 1706.4 = 141.95$ грн./год.

Витрати на оплату машинного часу:

I. $С_М = 141.95* 1168 = 165797.6$ грн.

II. $С_М = 141.95 * 1482.496 = 210440.31$ грн.

Накладні витрати складають 67% від заробітньої плати:

I. $С_Н = 165797.6 * 0.67 = 111084.39$ грн.

II. $С_Н = 210440.31 * 0.67 = 140995.01$ грн.

Вартість розробки програмного продукту:

I. $С_{ПП} = 233605 + 51393 + 165797.6 + 111084.39 = 561879.99$ грн.

II. $С_{ПП} = 296505 + 65231 + 210440.31 + 140995.01 = 713171.32$ грн.

Розрахуємо коефіцієнт техніко-економічного рівня:

$К_{ТЕР1} = 7{,}287/ 561879.99 = 1.29 * 10^{-5}$;

$К_{ТЕР2} = 10/ 713171.32 = 1.40 * 10^{-5}$;

4.4 Висновки

Звідси можна робити висновок, що найбільш ефективним є другий варіант з коефіцієнтом техніко-економічного рівня $1.40 * 10^{-5}$.

З альтернатив, що залишилися після відбору, було отримано два варіанти. Доцільним є використання першого варіанту реалізації продукту.

Цьому варіанту відповідає мова Java, власна математична бібліотека та існуючий алгоритм. Таким чином ми отримали універсальний засіб створення системи підтримки прийняття рішень для аналізу ринкових фінансових ризиків.

5 ВИСНОВКИ

У даній роботі були розглянуті деякі математичні поняття. А саме елементи функціонального аналізу, теорії багатовимірних відображень та теорії ігор. Головною метою було опис методу дослідження диференціальної гри. Для цього було наведено доведення важливої теореми для розглянутого методу. Крім цього був розроблений програмний продукт для моделювання та подальшого дослідження диференціальних ігор.

ПЕРЛІК ПОСИЛАНЬ

1. Чикрий А.А. Конфликтно управляемые процессы / Ин-т кибернетики им. В. М. Глушкова. – Киев: наук думка, 1992. – 384с. – ISBN 5-12-003426-8

2. Колмогоров А.Н., Фомин С.В. Элементы теории функций и функционального анализа. – Москва: Наука, 1981. – 544с.

3. Айзекс Р. Дифференциальные игры. – Москва: МИР – 1967 – 480с.

4. Григоренко Н. Л. Математические методы управления несколькими динамическими процессами, – Москва: Изд-во Моск. ун-та, 1990. – 197 с. – ISBN 5-211-00954-1.

5. Теория игр. URL: https://cito.mgsu.ru/COURSES/course753/media/337981978535638/pdf/teoria.pdf (дата звернення 30.06.2020)

ДОДАТОК А. ВИХІДНИЙ КОД ПРОГРАМНОГО ПРОДУКТУ

Block.java

```
import java.util.ArrayList;

public class Block {
    private int s;
    private double ro,sigma,alpha,betta;
    private ArrayList<Double> z1,z2,z3;



    public Block(int s, double ro, double sigma, double alpha, double betta,
ArrayList<Double> z1, ArrayList<Double> z2, ArrayList<Double> z3) {
        this.s = s;
        this.ro = ro;
        this.sigma = sigma;
        this.alpha = alpha;
        this.betta = betta;
        this.z1 = z1;
        this.z2 = z2;
        this.z3 = z3;
    }

    public int getS() {
        return s;
    }

    public double getRo() {
        return ro;
    }
```

```
public double getSigma() {
   return sigma;
}

public double getAlpha() {
   return alpha;
}

public double getBetta() {
   return betta;
}

public ArrayList<Double> getZ1() {
   return z1;
}

public ArrayList<Double> getZ2() {
   return z2;
}

public ArrayList<Double> getZ3() {
   return z3;
}

public void show(){
   System.out.println(s+" "+ro+" "+sigma+" "+alpha+" "+betta);
   System.out.println(z1);
   System.out.println(z2);
   System.out.println(z3);
}}
```

FileParser.java

```
import java.io.*;
import java.util.ArrayList;
import java.util.Scanner;

public class FileParser {
    private static Scanner scanner;
    private static boolean inflag = false;

    public static void _init_input(String filename) throws FileNotFoundException {
        if (inflag) return;
        scanner = new Scanner(new File(filename));
        inflag = true;
    }

    public static void _close_input() {
        if (inflag) {
            scanner.close();
            inflag = false;
        }
    }

    public static Block input() {
        if (!inflag) throw new IllegalMonitorStateException("No open stream");

        if (!scanner.hasNextDouble()) return null;
        int s = (int) scanner.nextDouble();
        if (s <= 0) throw new IllegalArgumentException("incorrect input data");
```

```
            ArrayList<Double> d = parse(scanner, 4);
            if (d == null) return null;
            if (d.get(2) <= 0.0 || d.get(3) <= 0.0) throw new IllegalArgumentException("incorrect input data");
            ArrayList<Double> z1 = parse(scanner, s);
            if (z1 == null) return null;
            ArrayList<Double> z2 = parse(scanner, s);
            if (z2 == null) return null;
            ArrayList<Double> z3 = parse(scanner, s);
            if (z3 == null) return null;

            return new Block(s, d.get(0), d.get(1), d.get(2), d.get(3), z1, z2, z3);
        }

        public static ArrayList<Double> parse(Scanner s, int cap) {
            ArrayList<Double> res = new ArrayList<Double>();

            for (int i = 0; i < cap; i++) {
                if (!s.hasNextDouble()) return null;
                res.add(s.nextDouble());
            }

            return res;
        }

        public static void output(String filename, ArrayList<Double> time) throws IOException {
            FileWriter o=new FileWriter(filename);
            for (Double x : time) o.write (x.toString()+"\n");
            o.close();
```

```
    }
}
```

Main.java

```
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.ArrayList;

public class Main {
    public static void main(String[] args) {

        String fileinput="datain.txt";
        String fileoutput="dataout.txt";
        TimeFunction._initiate(0.00001,1.0,2.0);
        ArrayList<Double> time=new ArrayList<>();

        try {
            FileParser._init_input(fileinput);
        } catch (FileNotFoundException e) {
            System.out.println("FILE NOT FOUND: "+fileinput);
            System.exit(-1);
        }


        Block b=FileParser.input();
        if(b==null) System.out.println(1);
        while(b!=null){
            time.add(TimeFunction.giveTime(b));
            b=FileParser.input();
        }
```

```
            FileParser._close_input();

            try {
                FileParser.output(fileoutput,time);
            } catch (IOException e) {
                System.out.println("UNABLE TO SAVE IN FILE: "+ fileoutput);
                System.out.println("Your data: ");
                for (double x: time ) System.out.println(x);
            }
            System.out.println("Calculated successfully");
        }
    }
```

TimeFunction.java

```
    import java.util.ArrayList;

    public class TimeFunction {

        private static double eps = 0.0,d=512.0,div_d=8.0;

        public static void _initiate(double epselon,double delta, double div_delta)
{
            if(epselon <= 0.0 || delta<=0 || div_delta<=0)throw new
IllegalArgumentException("unable init");
            eps = epselon;
            d=delta;
            div_d=div_delta;
        }

        public static double giveTime(Block b) {
```

```
            if (!_isInit_()) throw new IllegalStateException("Error init Time function");
            if (!_check_(b.getS(), b.getZ1(), b.getZ2(), b.getZ3(), b.getAlpha(), b.getBetta()))
                throw new IllegalArgumentException("Time function error");

            double t = 0.0;
            double dif=diff(b,t);
            do{
                t=logic(b,t);
                dif=diff(b,t);
                d/=div_d;
            } while(dif<0 || dif>eps);
            return t;
        }

        private static boolean _isInit_() {
            return (eps > 0.0) && (eps <= 0.1);
        }

        private static boolean _check_(int s, ArrayList<Double> z1, ArrayList<Double> z2, ArrayList<Double> z3,
                                       double alpha, double betta) {

            if (z1 == null || z2 == null || z3 == null) return false;

            return (alpha > 0) && (betta > 0) && (s == z1.size()) && (s == z2.size()) && (s == z3.size());
        }
```

```
private static double logic(Block b, double t0){
    double t=t0;
    while (diff(b,t)<eps) t+=d;
    return t-d;
}

private static double diff(Block b,double t){
    return rightNum(b.getRo(), b.getSigma(), b.getAlpha(), b.getBetta(), t)
        - leftNum(b.getS(), b.getZ1(), b.getZ2(), b.getZ3(), b.getAlpha(),
b.getBetta(), t);
}

private static double exp(double x) {
    double e;
    e = 2.71828182845904523536028747135 27;
    return Math.pow(e, x);
}

private static double expm(double t, double a) {
    return (1.0 - exp((-1) * a * t));
}

private static double mexp(double t, double a) {
    return (exp((-1) * a * t) - 1.0);
}

private static double rightNum(double r, double sigma, double a, double b,
double t) {
    return Math.abs((r / a - sigma / b) * t + expm(t, a) + mexp(t, b));
}
```

```
        private static ArrayList<Double> vector(int s, ArrayList<Double> z1,
ArrayList<Double> z2, ArrayList<Double> z3,
                                    double alpha, double betta, double t) {
            ArrayList<Double> result = new ArrayList<Double>();

            for (int i = 0; i < s; i++) {
                result.add(z1.get(i) + expm(t, alpha) * z2.get(i) + mexp(t, betta) *
z3.get(i));
            }

            return result;
        }

        private static double norma(ArrayList<Double> x) {
            double result = 0.0;
            for (Double i : x) result += i*i;
            return Math.sqrt(result);
        }

        private static double leftNum(int s, ArrayList<Double> z1,
ArrayList<Double> z2, ArrayList<Double> z3,
                                double alpha, double betta, double t) {
            return norma(vector(s, z1, z2, z3, alpha, betta, t));
        }
    }
```

ДОДАТОК Б. ПРИКЛАДИ РОБОЧИХ ФАЙЛІВ

datain.txt

```
3,0
5,0 3,0 1,0 1,0
1,0 2,0 3,0
4,0 5,0 6,0
7,0 8,0 9,0
4,0
2,0 1,0 1,0 1,0
1,0 2,0 3,0 0,123
4,0 5,0 6,0 23,44
7,0 8,0 9,0 22,0
```

dataout.txt

```
0.7787990570068359
2.36257266998291
```

ДОДАТОК В. ПРЕЗЕНТАЦІЯ

Дипломна робота на тему «Перший прямий метод Л. С. Понтрягіна в диференціальній грі»

ШПИЛЬОВОГО ІЛЛІ ВЯЧЕСЛАВОВИЧА

Диференціальні ігри переслідування-тікання

$\dot{z} = Az + \varphi(u, v), z \in \mathbb{R}^n, u \in U, v \in V$

$U, V \in K(\mathbb{R}^n)$

$\varphi: U \times V \to \mathbb{R}^n$

$M^* = M^0 + M$

Метод Понтрягіна

Початковий стан позначимо z^0

$\varphi(v) = \{\varphi(u, v) \mid u \in U\}, v \in V$ $\qquad$ $P(z) = \inf\left\{t \geq 0 \mid \pi e^{At} z \in M - \int_0^t W(x)dx\right\}$

$W(t, v) = \pi e^{At}\varphi(v)$

$W(t) = \bigcap_{v \in V} W(t, v)\,, t \geq 0$

$\forall t \geq 0 : W(t) \neq \emptyset$

$U(t, v) = \left\{u \in U \mid \pi e^{A(P(z^0)-t)}\varphi(u, v) - \gamma(P(z^0) - t)\right\}, t \in [0, P(z^0)], v \in V$

Контрольний приклад Понтрягіна

$\ddot{x} + a\dot{x} = ru, x \in \mathbb{R}^s, \|u\| \leq 1, a, r > 0$

$\ddot{y} + b\dot{y} = cv, y \in \mathbb{R}^s, \|v\| \leq 1, b, c > 0$

$u \in U, v \in V$

$z = column(z_1, z_2, z_3)$

$z_1, z_2, z_3 \in \mathbb{R}^n$

$z_1 = x - y$

$z_2 = \dot{x}$

$z_3 = \dot{y}$

$P(z^0) = \min\left\{t \geq 0 \mid \left\|z_1^0 + \frac{1-e^{-at}}{a} z_2^0 - \frac{1-e^{-bt}}{b} z_3^0\right\| \leq\leq \left|\left(\frac{r}{a} - \frac{c}{b}\right)t + (1 - e^{-at}) - (1 - e^{-bt})\right|\right\}$

$u^*(t) = v(t)$

Програмний продукт

datain.txt

3,0

5,0 3,0 1,0 1,0

1,0 2,0 3,0

4,0 5,0 6,0

7,0 8,0 9,0

4,0

2,0 1,0 1,0 1,0

1,0 2,0 3,0 0,123

4,0 5,0 6,0 23,44

7,0 8,0 9,0 22,0

dataout.txt

0.7787990570068359

2.36257266998291

Дякую за увагу